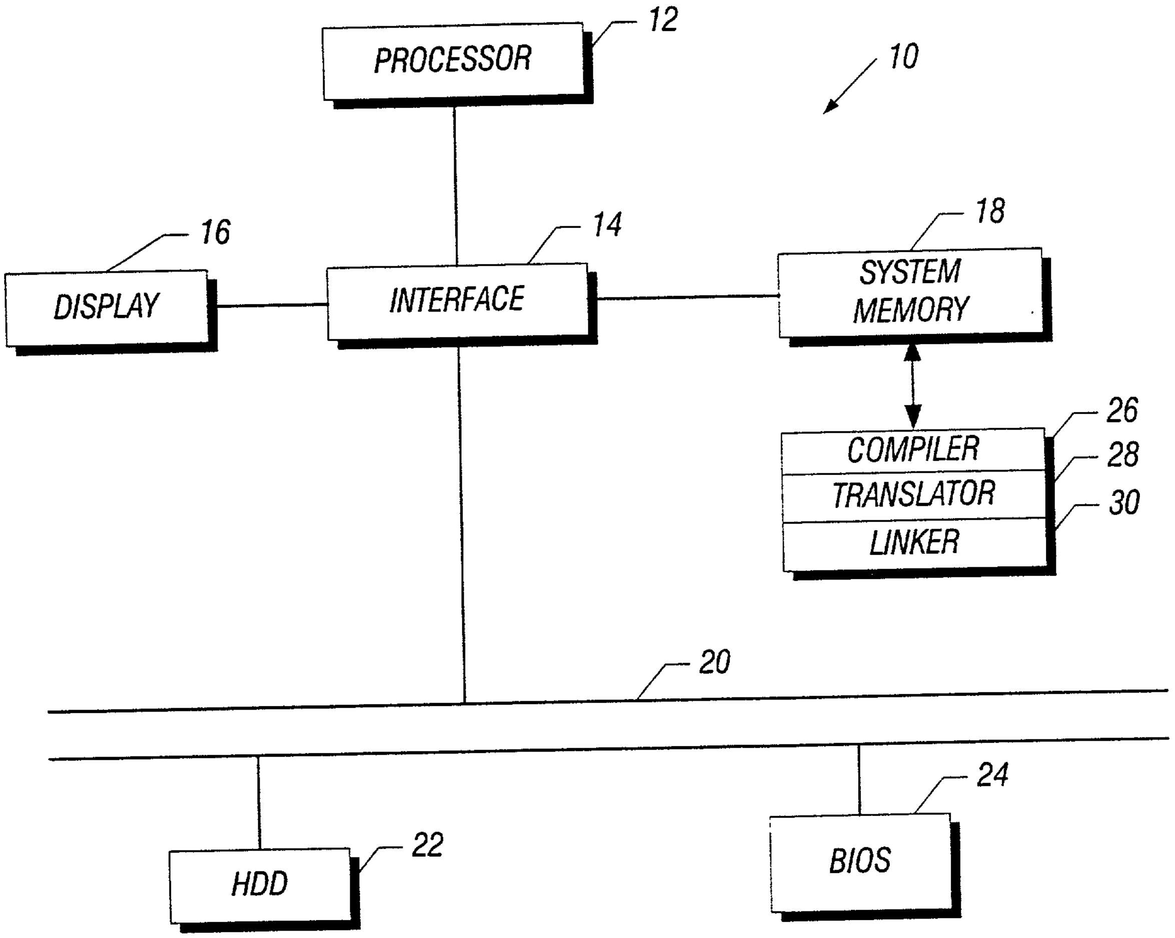

PROCESSOR
12
10
DISPLAY
16
INTERFACE
14
SYSTEM MEMORY
18
COMPILER
26
TRANSLATOR
28
LINKER
30
20
HDD
22
BIOS
24

FIG. 1

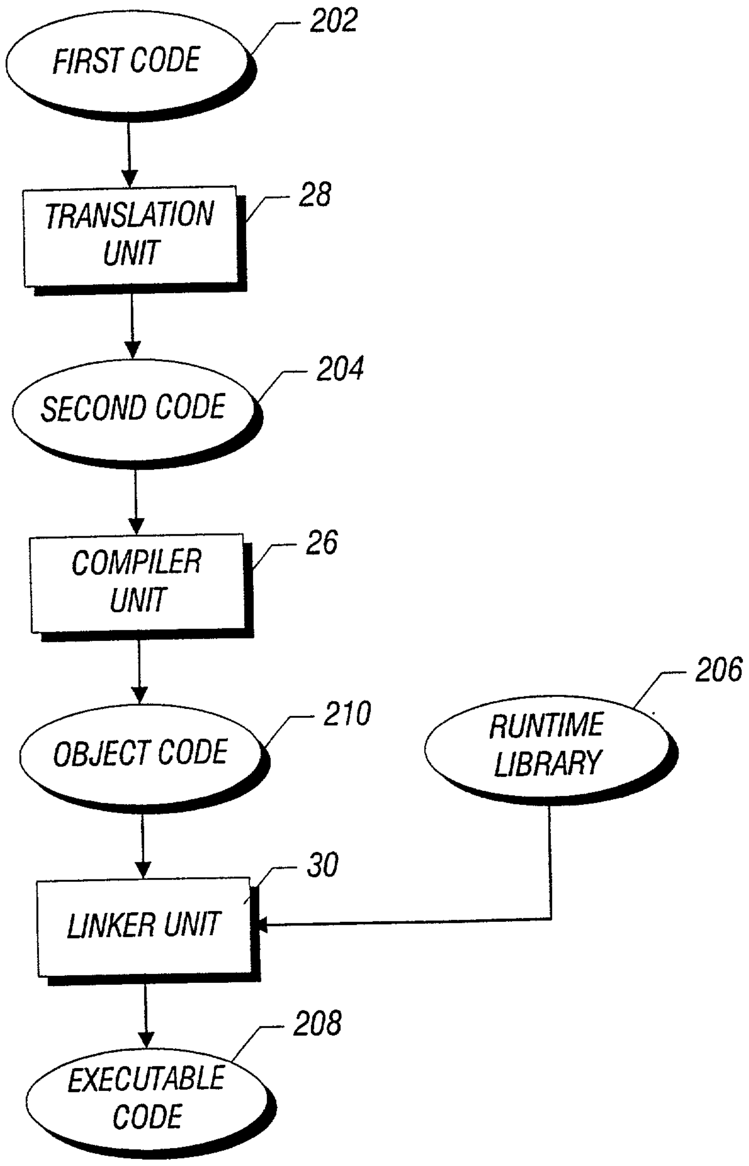

FIRST CODE
202
TRANSLATION UNIT
28
SECOND CODE
204
COMPILER UNIT
26
OBJECT CODE
210
RUNTIME LIBRARY
206
LINKER UNIT
30
EXECUTABLE CODE
208

FIG. 2

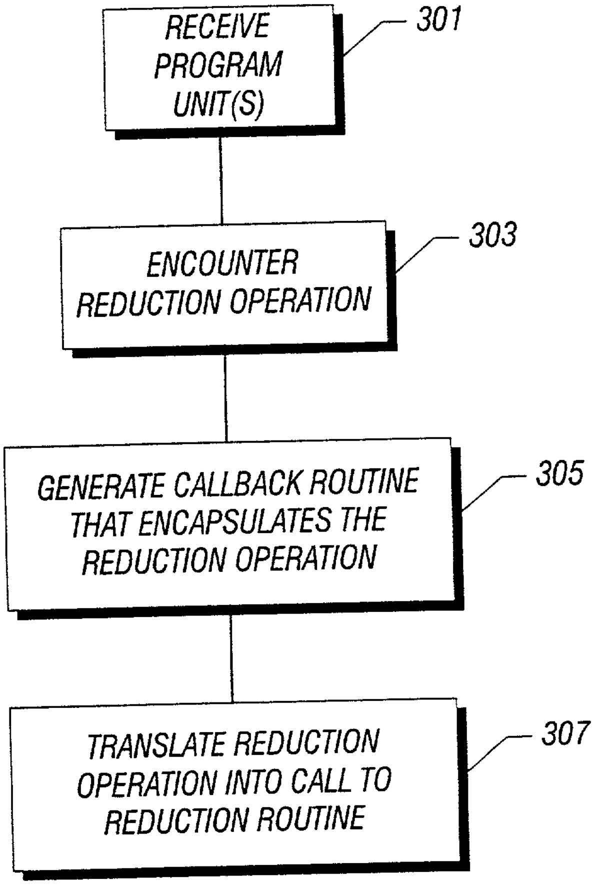

RECEIVE PROGRAM UNIT(S)
301
ENCOUNTER REDUCTION OPERATION
303
GENERATE CALLBACK ROUTINE THAT ENCAPSULATES THE REDUCTION OPERATION
305
TRANSLATE REDUCTION OPERATION INTO CALL TO REDUCTION ROUTINE
307

FIG. 3

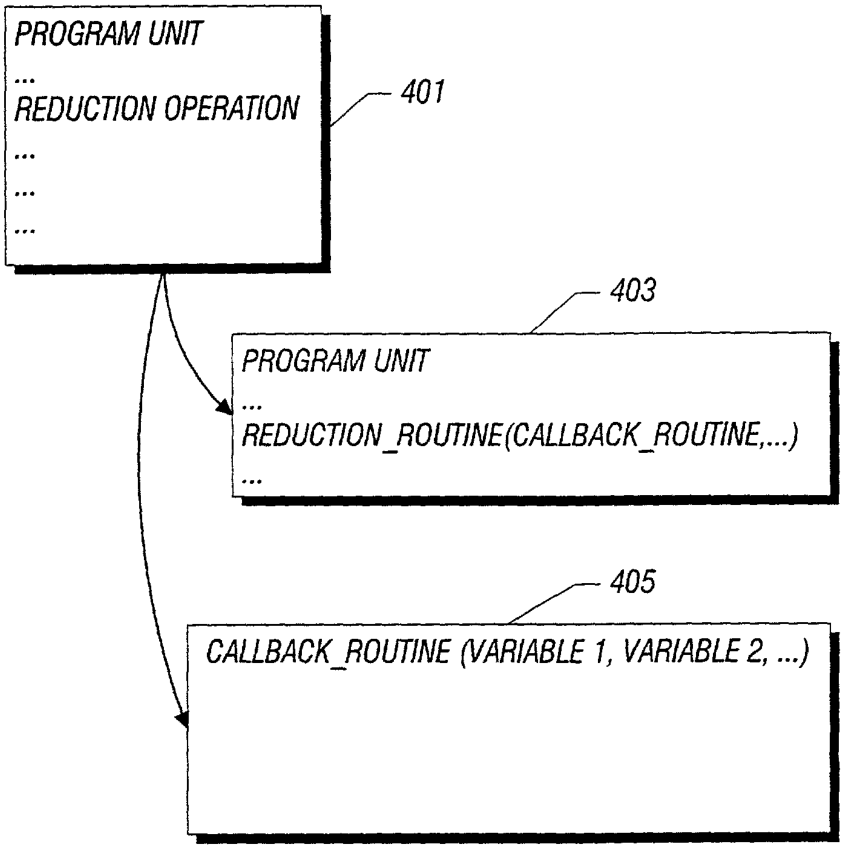
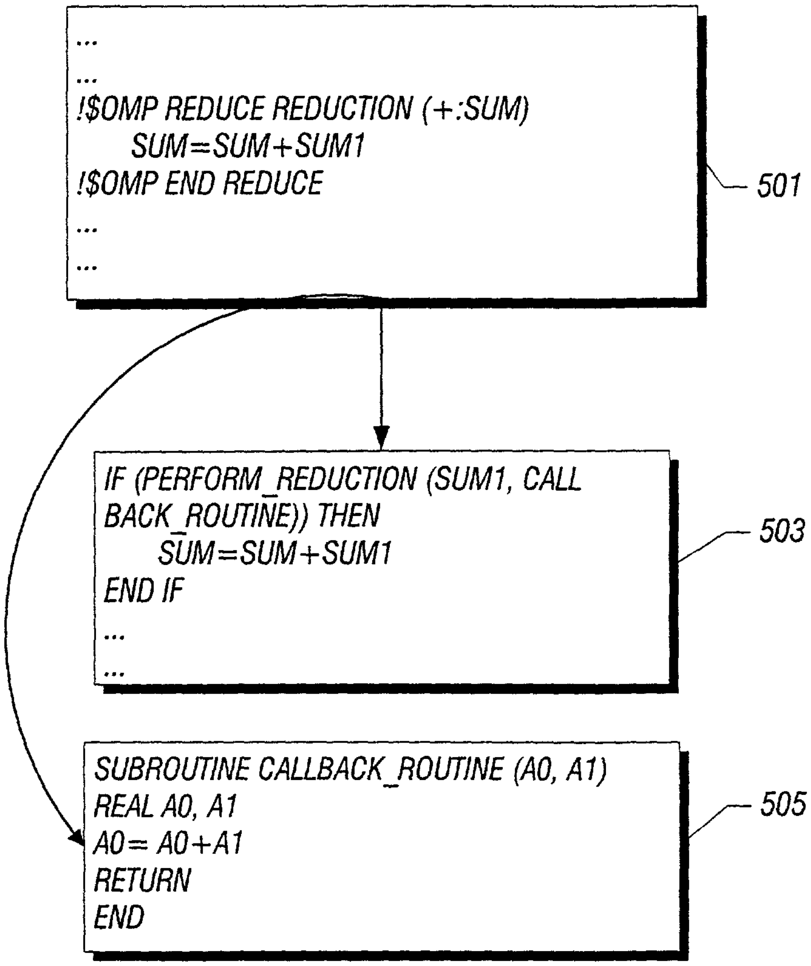

PROGRAM UNIT
...
REDUCTION OPERATION
...
...
...
401
403
PROGRAM UNIT
...
REDUCTION_ROUTINE(CALLBACK_ROUTINE,...)
...
405
CALLBACK_ROUTINE (VARIABLE 1, VARIABLE 2, ...)

FIG. 4

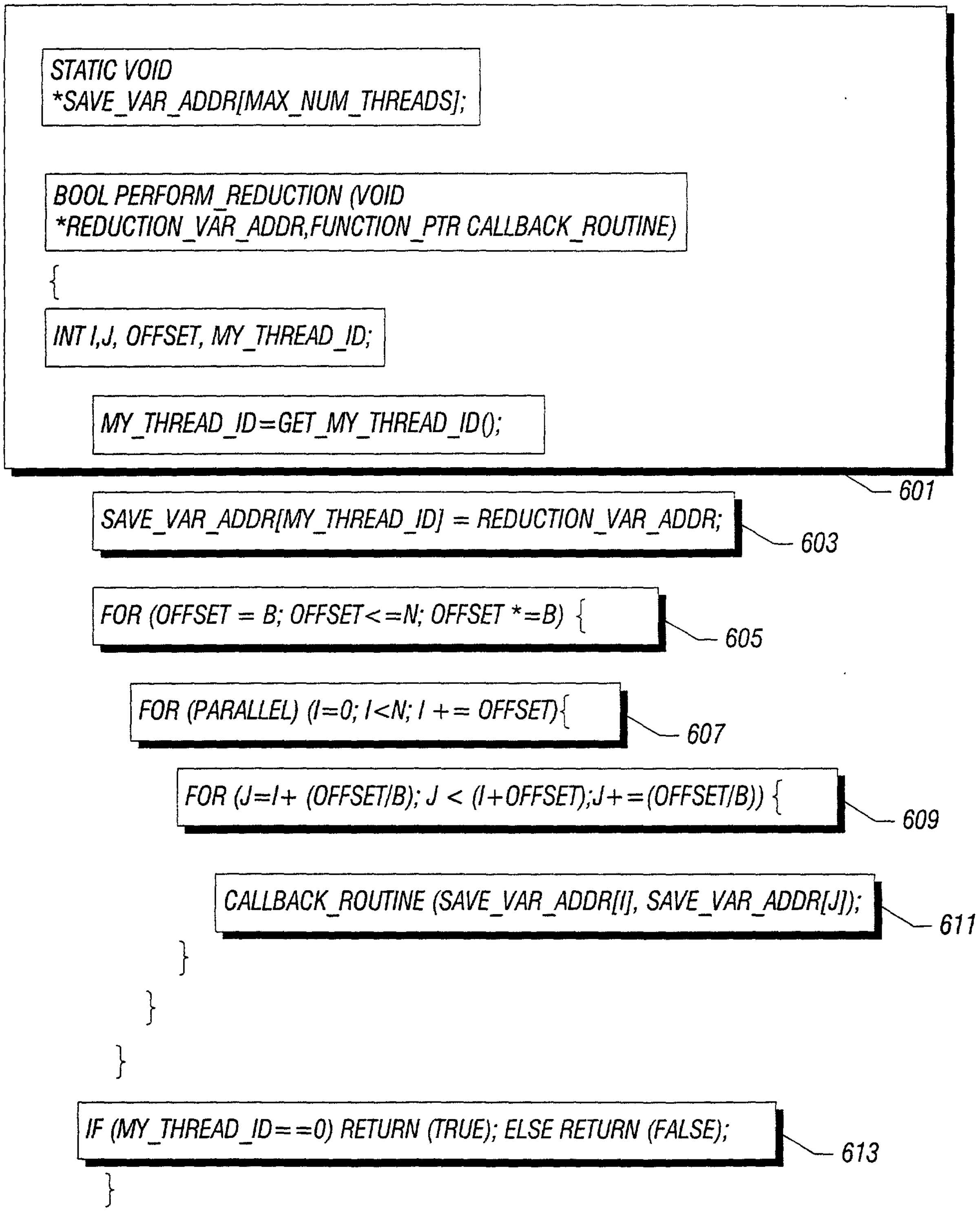


```
...
...
!$OMP REDUCE REDUCTION (+:SUM)
      SUM=SUM+SUM1
!$OMP END REDUCE
...
...
```

```
IF (PERFORM_REDUCTION (SUM1, CALL
BACK_ROUTINE)) THEN
      SUM=SUM+SUM1
END IF
...
...
```

```
SUBROUTINE CALLBACK_ROUTINE (A0, A1)
REAL A0, A1
A0= A0+A1
RETURN
END
```

FIG. 5



```
STATIC VOID
*SAVE_VAR_ADDR[MAX_NUM_THREADS];

BOOL PERFORM_REDUCTION (VOID
*REDUCTION_VAR_ADDR,FUNCTION_PTR CALLBACK_ROUTINE)
{
INT I,J, OFFSET, MY_THREAD_ID;

    MY_THREAD_ID=GET_MY_THREAD_ID();

    SAVE_VAR_ADDR[MY_THREAD_ID] = REDUCTION_VAR_ADDR;

    FOR (OFFSET = B; OFFSET<=N; OFFSET *=B) {

        FOR (PARALLEL) (I=0; I<N; I += OFFSET){

            FOR (J=I+ (OFFSET/B); J < (I+OFFSET);J+=(OFFSET/B)) {

                CALLBACK_ROUTINE (SAVE_VAR_ADDR[I], SAVE_VAR_ADDR[J]);
            }
        }
    }
   IF (MY_THREAD_ID==0) RETURN (TRUE); ELSE RETURN (FALSE);
   }
```

FIG. 6

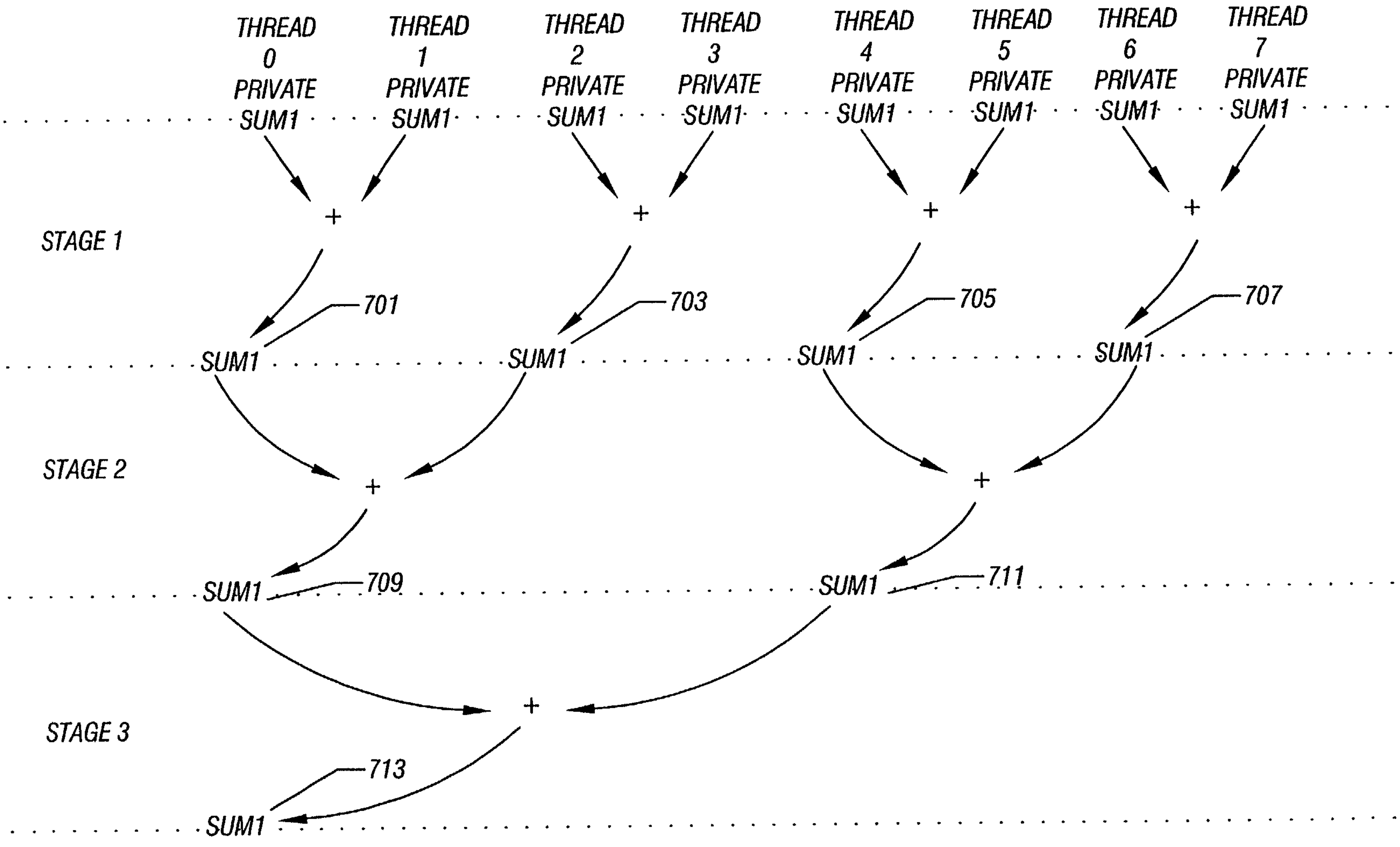
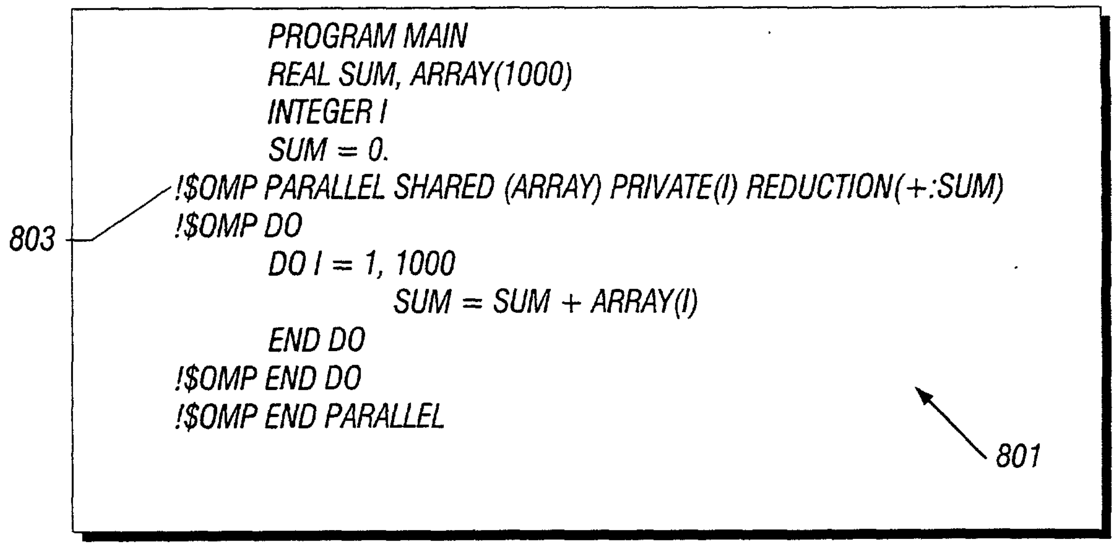
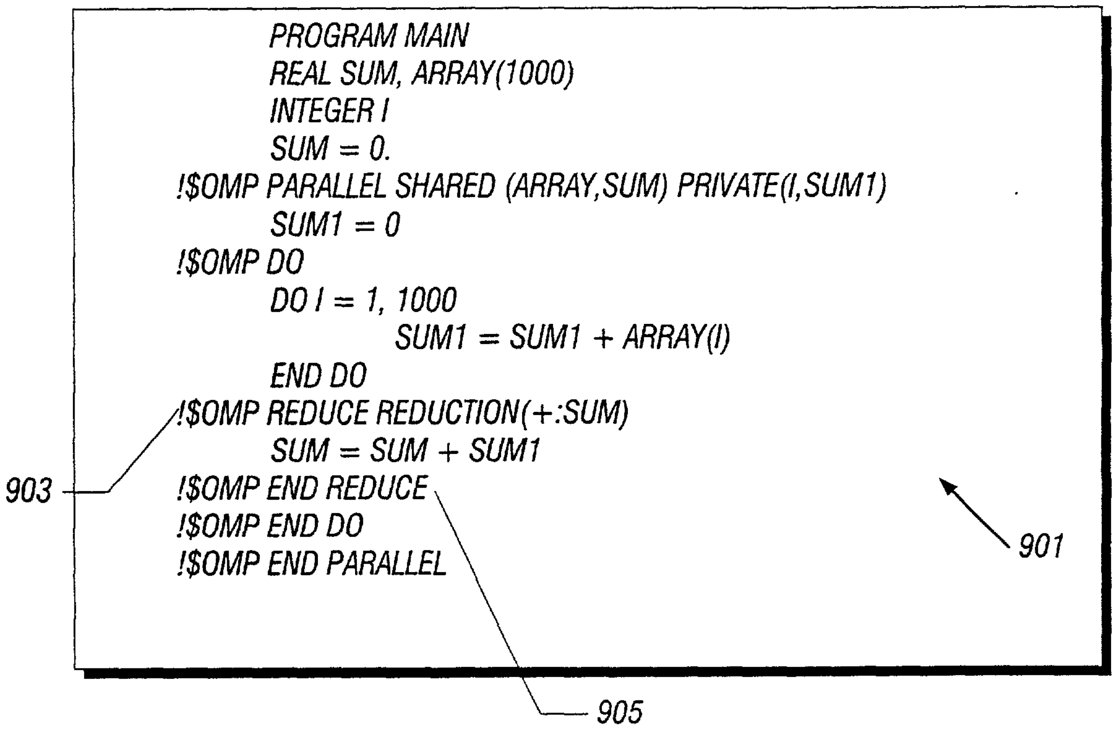

THREAD 0 PRIVATE SUM1
THREAD 1 PRIVATE SUM1
THREAD 2 PRIVATE SUM1
THREAD 3 PRIVATE SUM1
THREAD 4 PRIVATE SUM1
THREAD 5 PRIVATE SUM1
THREAD 6 PRIVATE SUM1
THREAD 7 PRIVATE SUM1
STAGE 1
+
+
+
+
701
703
705
707
SUM1
SUM1
SUM1
SUM1
STAGE 2
+
+
SUM1
709
SUM1
711
STAGE 3
+
713
SUM1

FIG. 7



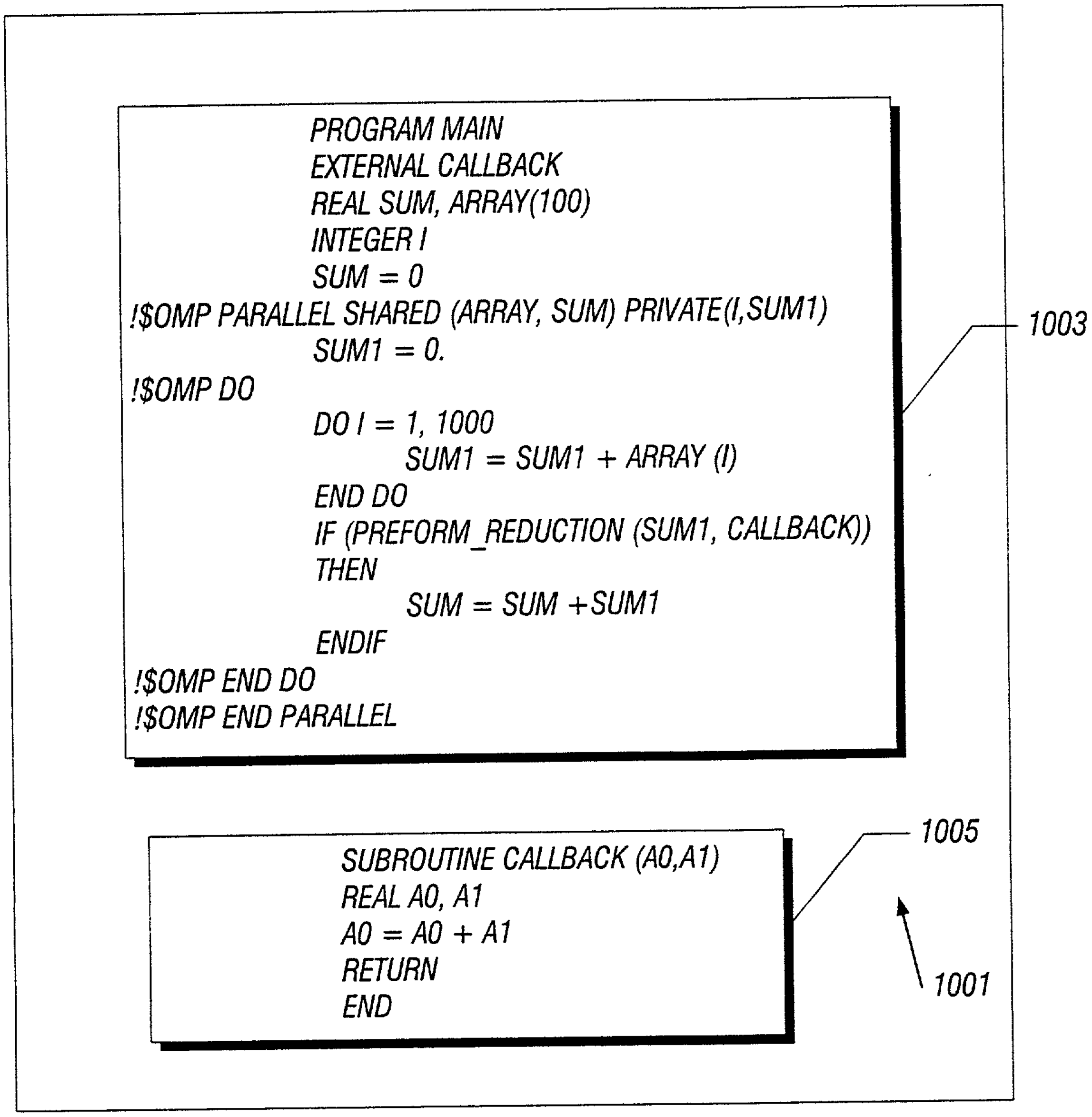
```
          PROGRAM MAIN
          REAL SUM, ARRAY(1000)
          INTEGER I
          SUM = 0.
!$OMP PARALLEL SHARED (ARRAY) PRIVATE(I) REDUCTION(+:SUM)
!$OMP DO
          DO I = 1, 1000
                    SUM = SUM + ARRAY(I)
          END DO
!$OMP END DO
!$OMP END PARALLEL
```

FIG. 8



```
          PROGRAM MAIN
          REAL SUM, ARRAY(1000)
          INTEGER I
          SUM = 0.
!$OMP PARALLEL SHARED (ARRAY,SUM) PRIVATE(I,SUM1)
          SUM1 = 0
!$OMP DO
          DO I = 1, 1000
                    SUM1 = SUM1 + ARRAY(I)
          END DO
!$OMP REDUCE REDUCTION(+:SUM)
          SUM = SUM + SUM1
!$OMP END REDUCE
!$OMP END DO
!$OMP END PARALLEL
```

FIG. 9



```
              PROGRAM MAIN
              EXTERNAL CALLBACK
              REAL SUM, ARRAY(100)
              INTEGER I
              SUM = 0
!$OMP PARALLEL SHARED (ARRAY, SUM) PRIVATE(I,SUM1)
              SUM1 = 0.
!$OMP DO
              DO I = 1, 1000
                     SUM1 = SUM1 + ARRAY (I)
              END DO
              IF (PREFORM_REDUCTION (SUM1, CALLBACK))
              THEN
                     SUM = SUM +SUM1
              ENDIF
!$OMP END DO
!$OMP END PARALLEL
```

```
              SUBROUTINE CALLBACK (A0,A1)
              REAL A0, A1
              A0 = A0 + A1
              RETURN
              END
```

FIG. 10